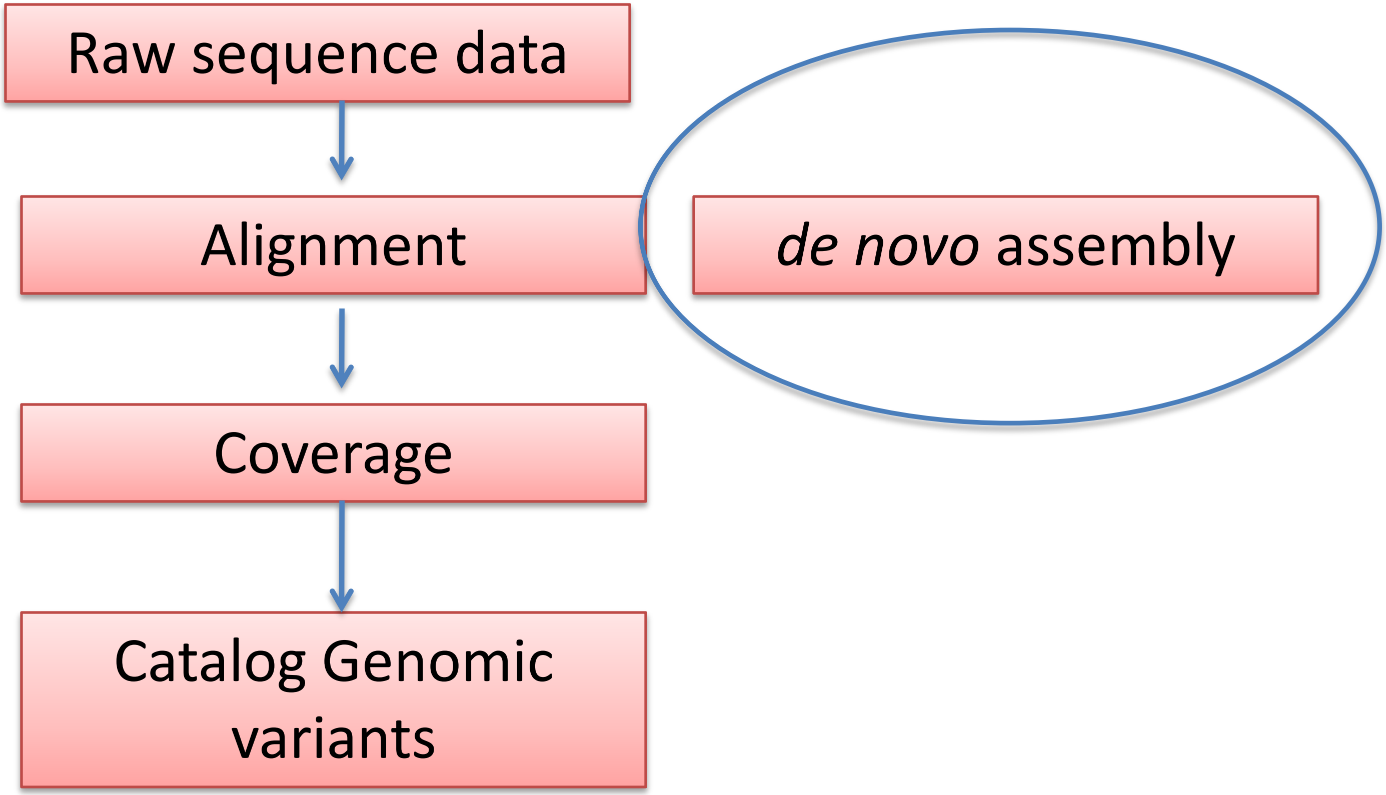
De novo assembly

London School of Hygiene and Tropical Medicine

Some aspects of the course



Raw sequence data

Alignment

de novo assembly

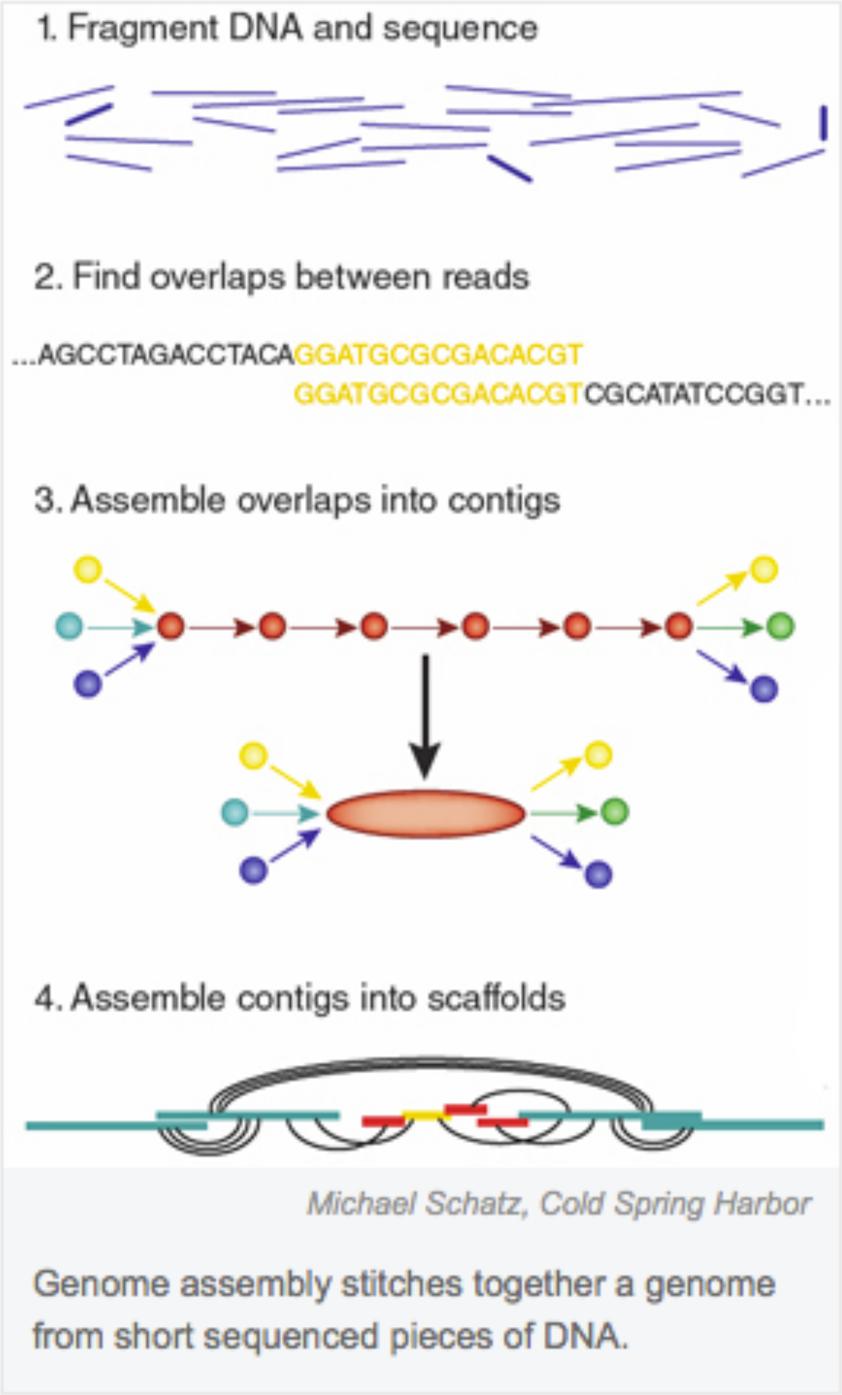
Coverage

Catalog Genomic variants

Population genetics

Whole genome Association studies

De novo assembly



1. Fragment DNA and sequence

2. Find overlaps between reads

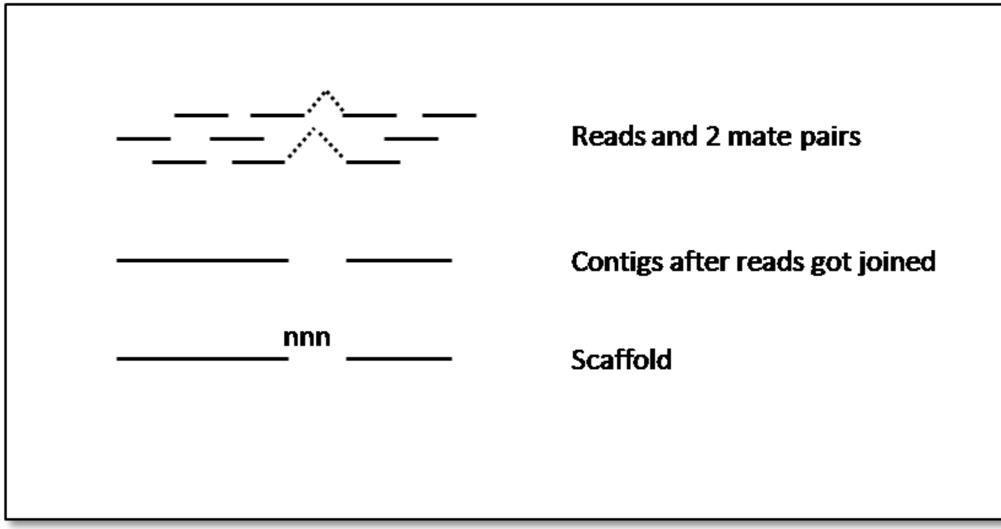
...AGCCTAGACCTACAGGATGCGCGACACGT
GGATGCGCGACACGTCGCATATCCGGT...

3. Assemble overlaps into contigs

4. Assemble contigs into scaffolds

Michael Schatz, Cold Spring Harbor

Genome assembly stitches together a genome from short sequenced pieces of DNA.

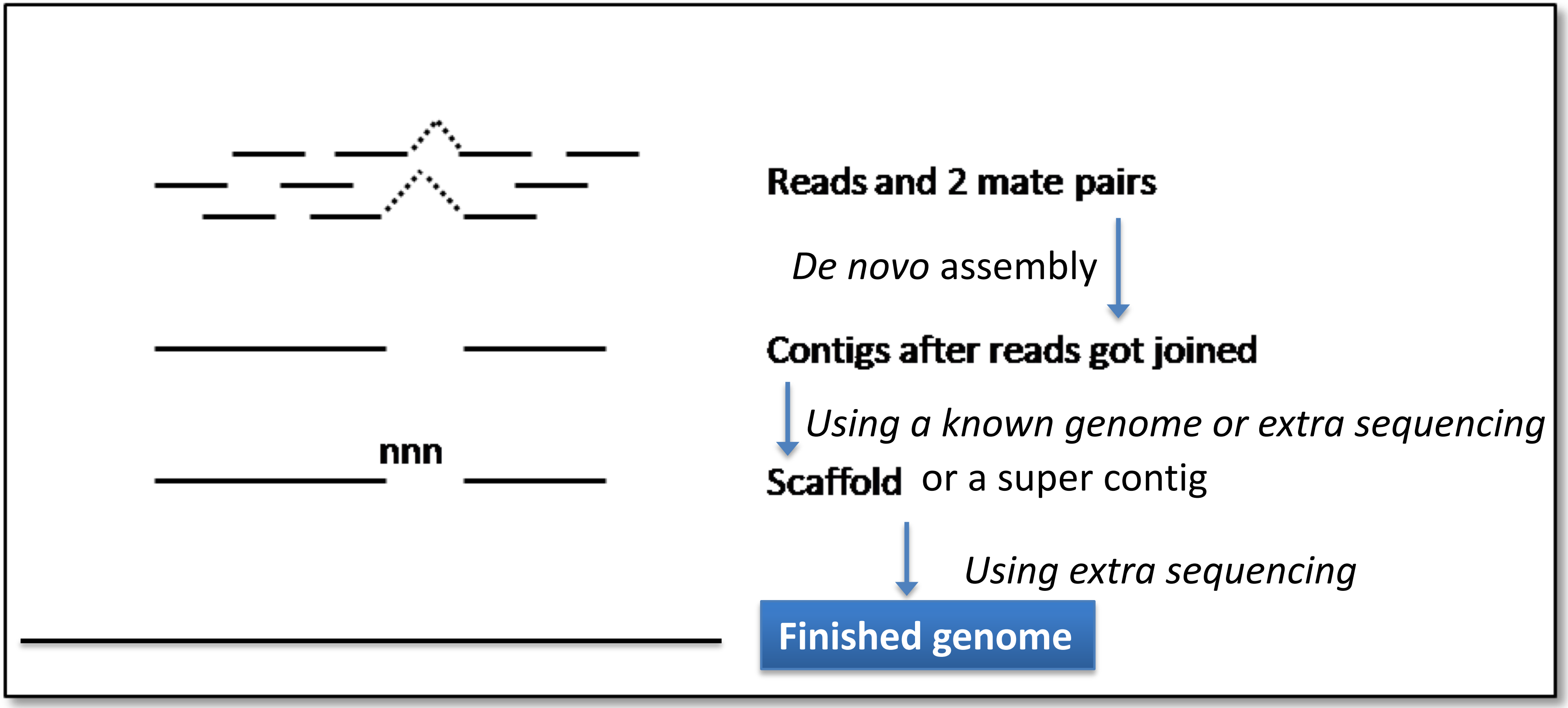


Reads and 2 mate pairs

Contigs after reads got joined

Scaffold

One ultimate goal is a “finished genome”



Reads and 2 mate pairs

De novo assembly

Contigs after reads got joined

Using a known genome or extra sequencing

Scaffold or a super contig

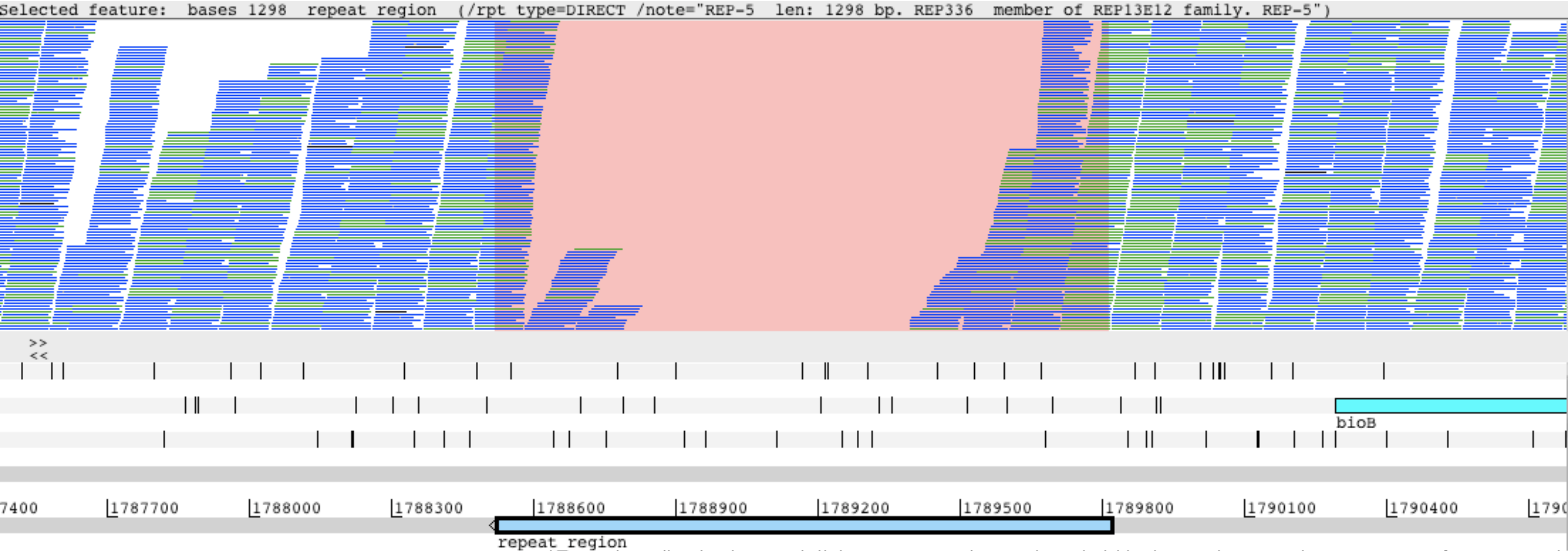
Using extra sequencing

Finished genome

Why assemble?

- Reference genomes unavailable
- Rapidly changing organisms
- Highly variable/unstable regions
- Investigate structural variants
  - Insertions (e.g. novel insertions)
  - Deletions
  - Repeat regions
  - Inversions
- *In silico* genotype (e.g. reconstruct a MIRU-VNTR)
- Novel transcripts in a transcriptome

M. tuberculosis repeat region poorly mapped

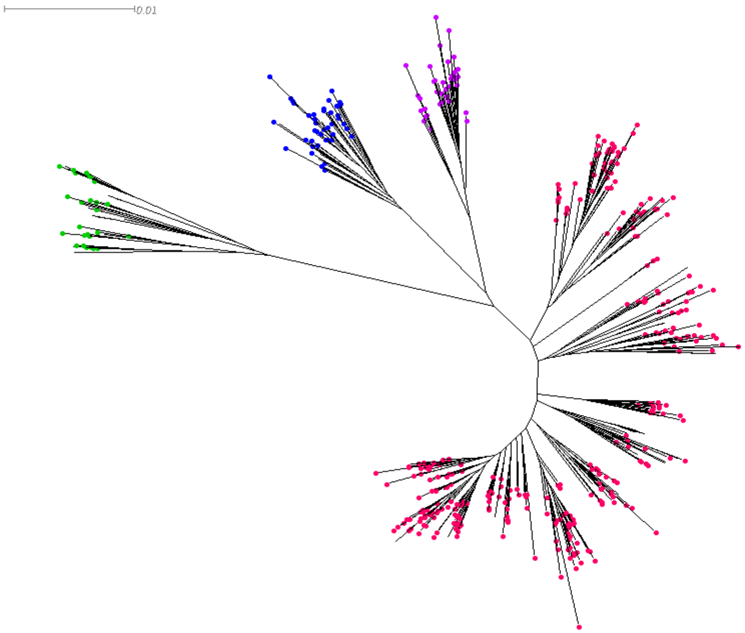


Selected feature: bases 1298 repeat region (/rpt type=DIRECT /note="REP-5 len: 1298 bp. REP336 member of REP13E12 family. REP-5")

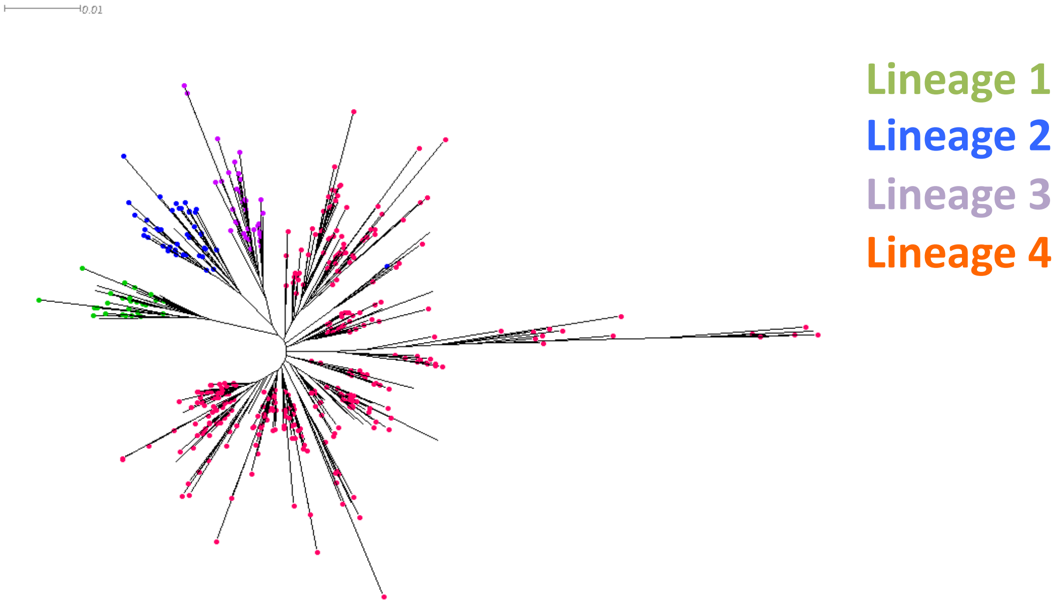
De novo assembly of reads that are single end mapped or unmapped to fill in gaps

De novo assembly of bacterial genomes e.g. *M. tuberculosis*

- Members of the PE/PPE family are thought to be virulence factors, which participate in evasion of the host immune response.
- Typically PE/PPE regions are excluded from analyses
- *De novo* assembled whole genomes, including PE/PPE genes across 518 clinical isolates (lineages 1-4).

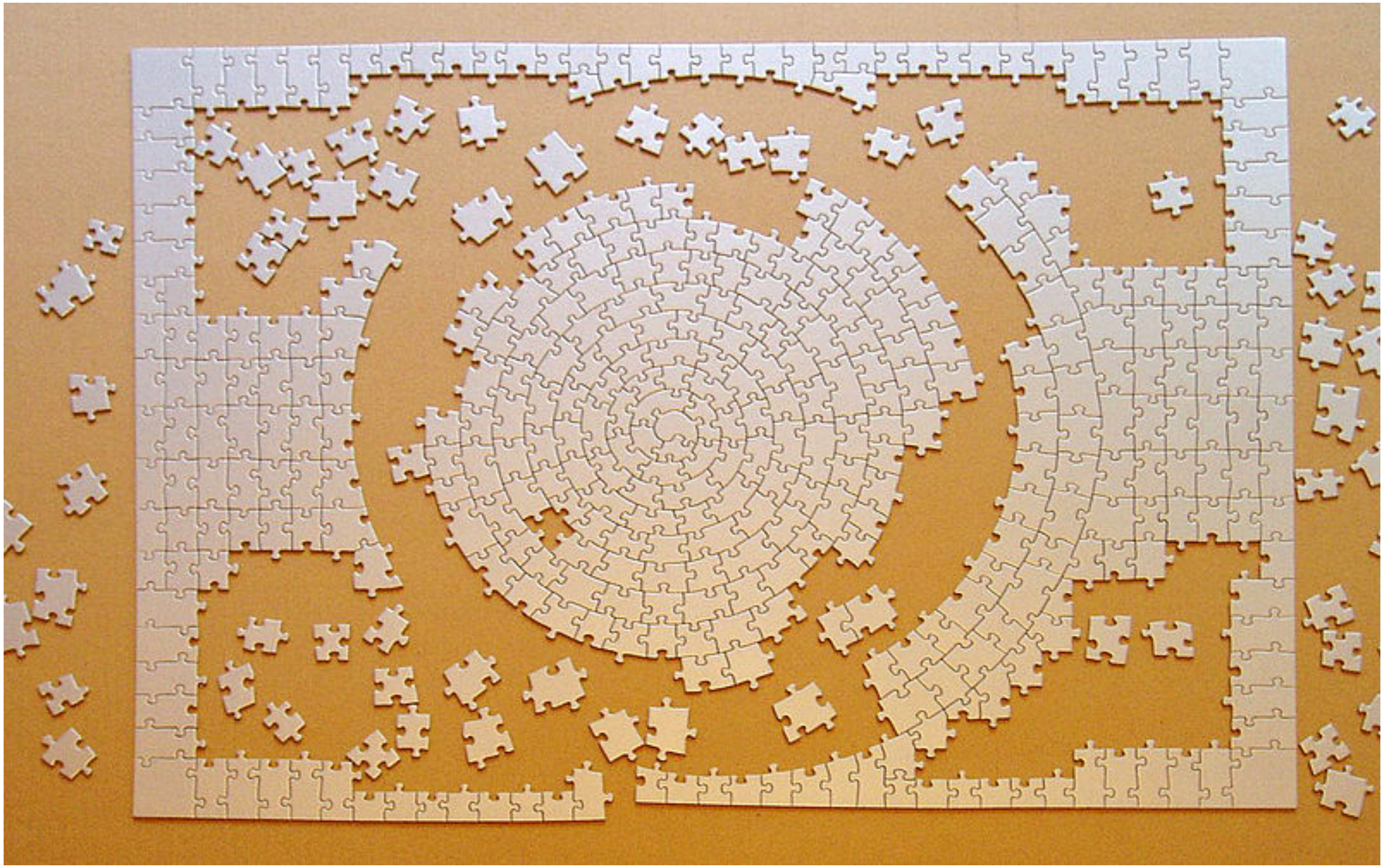


Whole genome assembly (68k SNPs)



PE/PPE region assembly

Constructing a genome can be viewed as a jigsaw puzzle



The genomic jigsaw puzzle

- A sample *genome* is a picture
- Each *short read* is a jigsaw piece
- *Mapping* uses a known picture close to the actual picture (a *reference*) to help placing reads
  - Bigger pieces (longer reads) make it easier to reconstruct picture
  - Knowing the approximate distance between two pieces *(paired-end reads*) makes it easier to place them
  - Damaged pieces (*sequencing errors*) make it difficult
  - Spotting the differences between genomes identifies *variants*

The genomic jigsaw puzzle

- *Assembly* creates the picture without the reference
  - No reference means inaccuracies in pictures (genomes) are not considered
  - No reference means that poor quality regions not assembled
  - More difficult to overcome hard to sequence regions
  - DNA sequence reads may fit together in more than one way because of repetitive sequences within the genome.
  - Methods aim to create the most complete reconstruction possible without introducing errors.
- For assembly and mapping
  - Additional information assists to improve genome characterisation (e.g. read length, paired-end reads)
  - Junk in junk out – quality control is important

Using mapping and assembly

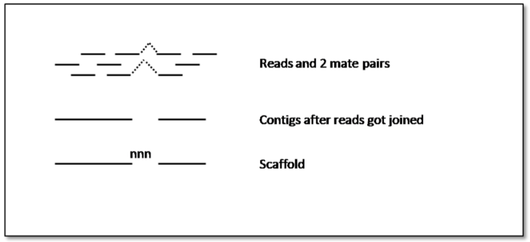
- Assembly is providing complementary information to mapping
- Map contigs to a reference or compare to another genome
  - Contigs are like long reads
  - Confirm small variants
  - Identify larger variants
- Reconstruct difficult to map genomic regions
- Filter reads for post-processing
  - Keeping only those reads in interesting contigs and performing mapping

Long read assembly

- Methods for “Sanger” reads where coverage is low
- Greedy – “Add reads together that have large overlaps” (e.g. Celera Assembler, ARACHNE, PCAP)
- Second generation long reads (e.g. 700bp 454 technology) – need to account for sequencing errors
- Use Overlap/Layout/Consensus (OLC) approaches (e.g. Newbler assembler)
- PacBio RS II has the longest read lengths (>10kb)
  - PacBio-only *de novo* assembly (OLC – HGAP algorithm).
  - Hybrid *de novo* assembly. Using a combination of PacBio and short read data
- Short read sequencing is much cheaper, and the larger number of reads requires other algorithms

The Problem

- Input
  - Sequence data is composed of many short reads (50 to 150+ base pairs)
  - Often these are *paired* together
- Output
  - Reads are joined together to form *contigs*
  - Contigs are joined to form *supercontigs* or *scaffolding*



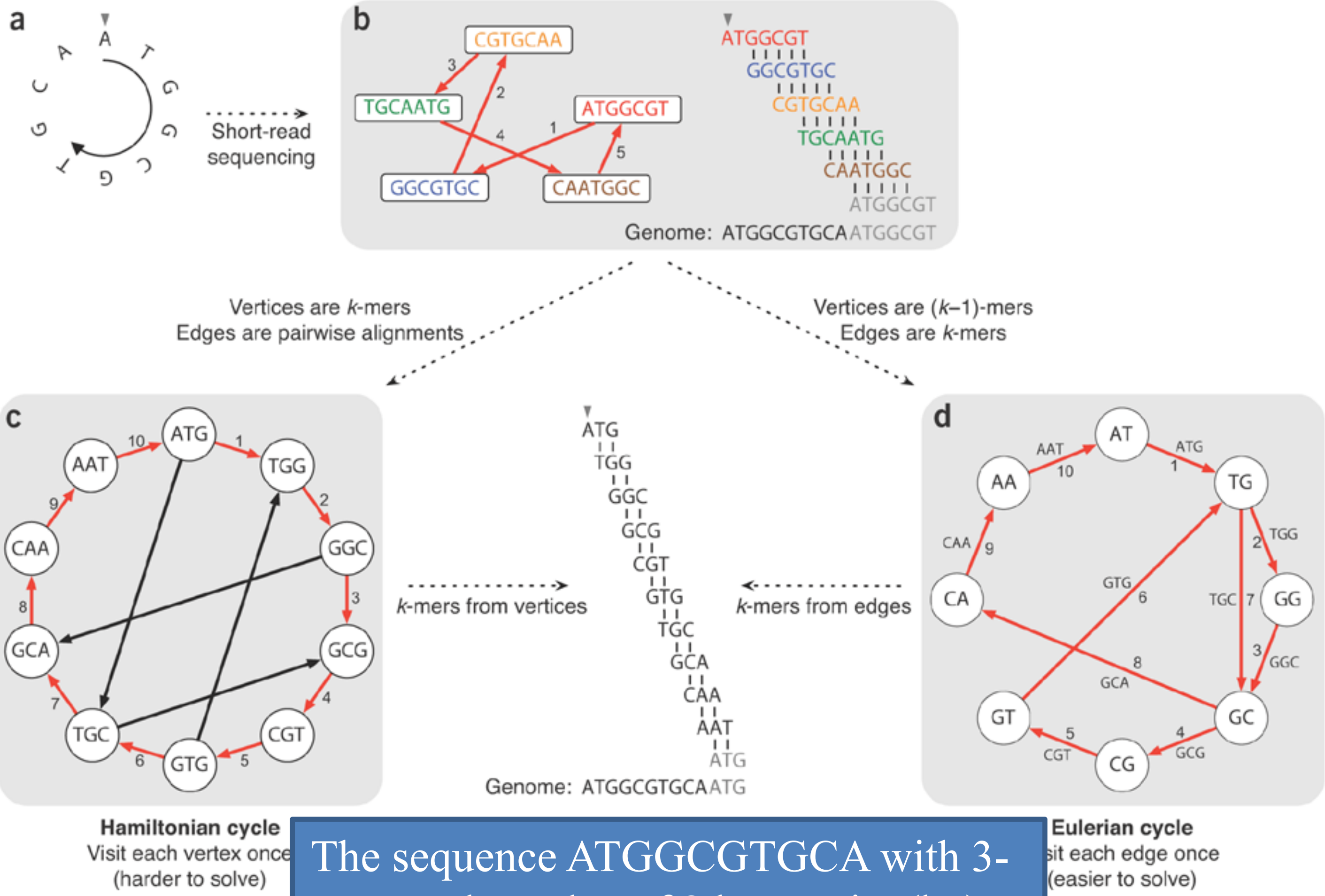
- Computationally intensive
  - More so than mapping
  - Potentially need many CPUs, large amounts of RAM
  - The solution involves a graph theory approach

Solutions to processing many short reads

- Greedy
  - Add reads together that have large overlaps
- Overlap/Layout/Consensus (OLC)
  - Nodes = Reads, Edges = Overlap
  - Good for long reads (200+ base pairs)
  - Slow
- De Bruijn Graphs
  - Nodes = k-mers, Edges = Overlap
  - Good for large datasets
  - Current *state of the art* technique

De Bruijn Graphs are used for short read data

By looking at contiguous subsets of short sequences (k'mers) we can construct graphs to describe links between reads



The sequence ATGGCGTGCA with 3-mers and overlap of 2 base pairs (bp).

The Plan

- What?
  - Joining short reads
- How?
  - A De Bruijn Graph theory approach implemented by specialist software (e.g. *velvet* software)
- Post-processing...
  - Use contigs to answer biological questions
  - Identify/verify structural variants
  - Create reference genomes

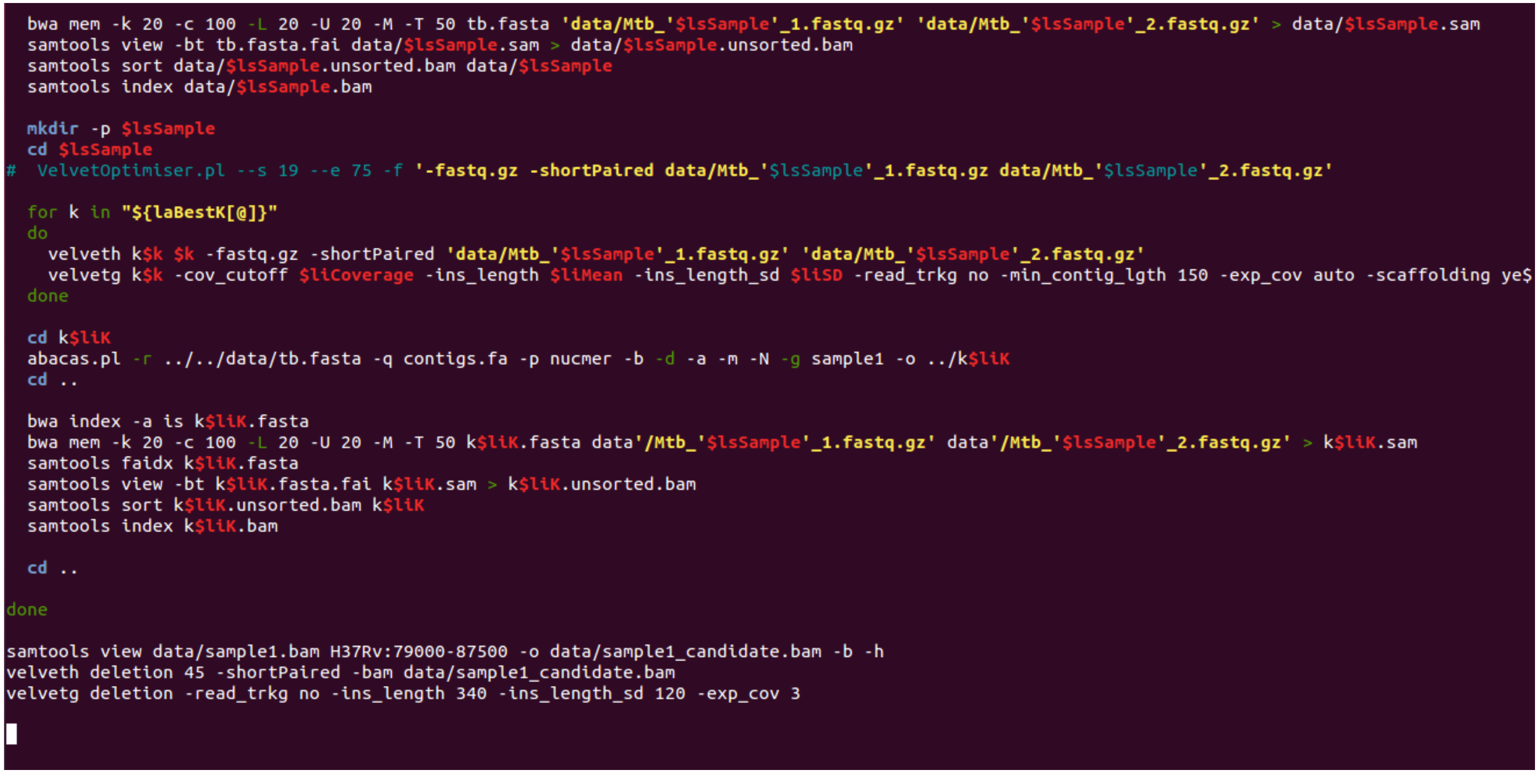
Evaluation of an assembly

In the absence of a high-quality reference genome, new genome assemblies are often evaluated on the basis of:

- the number of scaffolds and contigs required to represent the genome
- the proportion of reads that can be assembled
- the absolute length of contigs and scaffolds
- the length of contigs and scaffolds relative to the size of the genome
- The most commonly used metric is N50, the smallest scaffold or contig above which 50% of an assembly would be represented.

“Final graph has 978 nodes and n50 of 10508, max 54529, total 1374552, using 1397134/1510408 reads.”

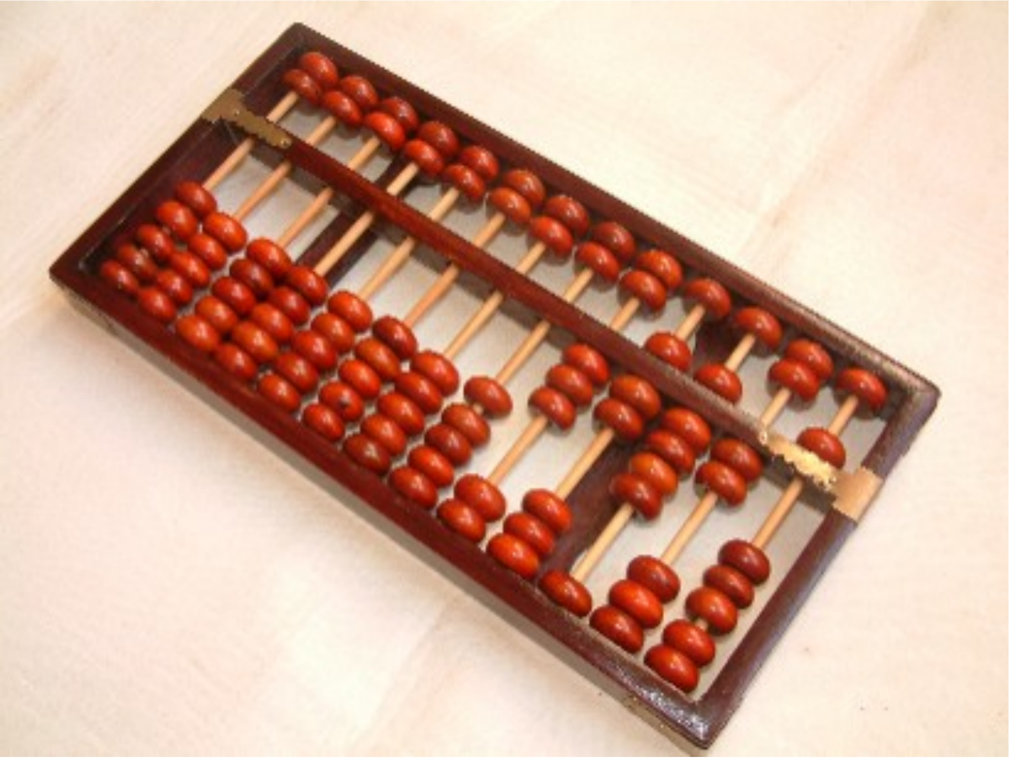
De Novo Assembly using Velvet



```
  bwa mem -k 20 -c 100 -L 20 -U 20 -M -T 50 tb.fasta 'data/Mtb_'$lsSample'_1.fastq.gz' 'data/Mtb_'$lsSample'_2.fastq.gz' > data/$lsSample.sam
  samtools view -bt tb.fasta.fai data/$lsSample.sam > data/$lsSample.unsorted.bam
  samtools sort data/$lsSample.unsorted.bam data/$lsSample
  samtools index data/$lsSample.bam

  mkdir -p $lsSample
  cd $lsSample
#  VelvetOptimiser.pl --s 19 --e 75 -f '-fastq.gz -shortPaired data/Mtb_'$lsSample'_1.fastq.gz data/Mtb_'$lsSample'_2.fastq.gz'

  for k in "${laBestK[@]}"
  do
    velveth k$k $k -fastq.gz -shortPaired 'data/Mtb_'$lsSample'_1.fastq.gz' 'data/Mtb_'$lsSample'_2.fastq.gz'
    velvetg k$k -cov_cutoff $liCoverage -ins_length $liMean -ins_length_sd $liSD -read_trkg no -min_contig_lgth 150 -exp_cov auto -scaffolding ye$
  done

  cd k$liK
  abacas.pl -r ../../data/tb.fasta -q contigs.fa -p nucmer -b -d -a -m -N -g sample1 -o ../k$liK
  cd ..

  bwa index -a is k$liK.fasta
  bwa mem -k 20 -c 100 -L 20 -U 20 -M -T 50 k$liK.fasta data'/Mtb_'$lsSample'_1.fastq.gz' data'/Mtb_'$lsSample'_2.fastq.gz' > k$liK.sam
  samtools faidx k$liK.fasta
  samtools view -bt k$liK.fasta.fai k$liK.sam > k$liK.unsorted.bam
  samtools sort k$liK.unsorted.bam k$liK
  samtools index k$liK.bam

  cd ..

done

samtools view data/sample1.bam H37Rv:79000-87500 -o data/sample1_candidate.bam -b -h
velveth deletion 45 -shortPaired -bam data/sample1_candidate.bam
velvetg deletion -read_trkg no -ins_length 340 -ins_length_sd 120 -exp_cov 3
```

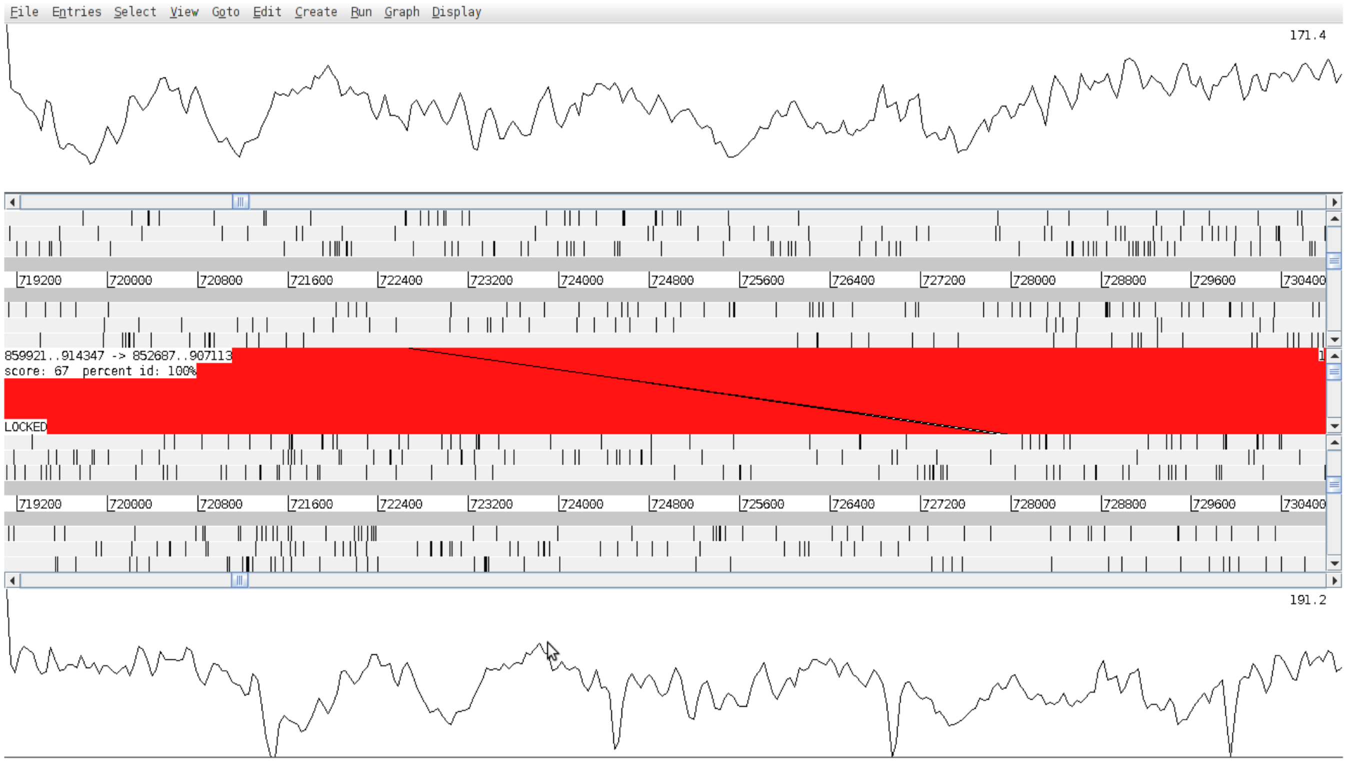
- Evaluation of the assembly using genomic coverage and summaries of contig lengths
- Many other approaches (e.g. *SPAdes*) – sometimes a compromise between accuracy and speed

Contig ordering



- Using a reference genome to order contigs (*e.g.* Abacas) and transfer annotations (*e.g. RATT, Prokka*)

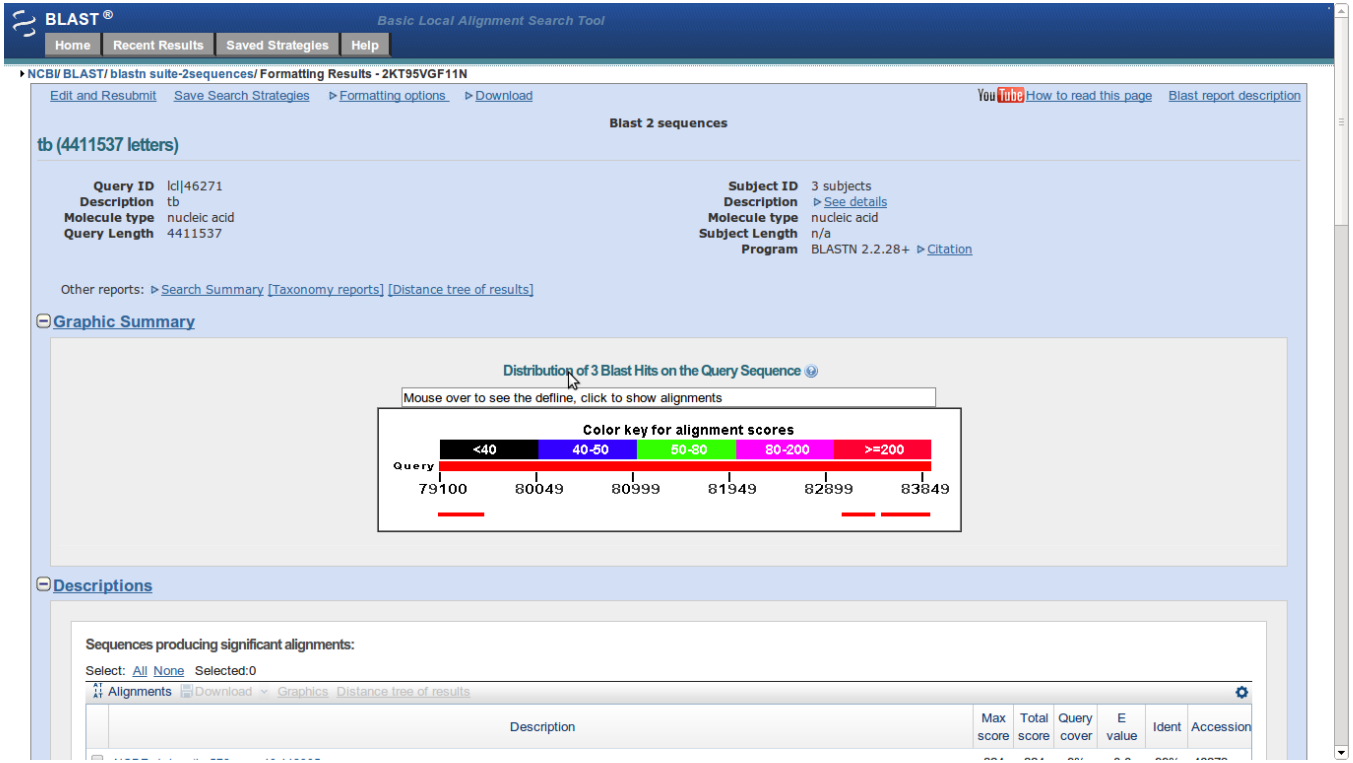
Mapping to the reference



859921..914347 -> 852687..907113
score: 67 percent id: 100%

- Visualisation of the assembly using the reference genome in Artemis ACT

Structural Variant Validation



Blast 2 sequences

tb (4411537 letters)

Query ID	lcl|46271	**Subject ID**	3 subjects
Description	tb	**Description**	See details
Molecule type	nucleic acid	**Molecule type**	nucleic acid
Query Length	4411537	**Subject Length**	n/a
		Program	BLASTN 2.2.28+ Citation

Other reports: Search Summary [Taxonomy reports] [Distance tree of results]

Graphic Summary

Distribution of 3 Blast Hits on the Query Sequence

Mouse over to see the defline, click to show alignments

Descriptions

Sequences producing significant alignments:

Select: All None Selected:0

	Description	Max score	Total score	Query cover	E value	Ident	Accession

- Using *blast* to validate structural variants